CSCI 5380

Network Virtualization and

Orchestration

Lab 2

OpenStack: Multi-tenants

University of Colorado Boulder

Department of Computer Science

Network Engineering

Professor Levi Perigo, Ph.D.

# **PART 1: OpenStack and multitenancy**

# Objective 1 - OpenStack: Overview

1. Explain the following components of OpenStack -
2. Nova
3. Swift
4. Cinder
5. Neutron
6. Glance
7. Keystone
8. Horizon
9. What is the difference between Users and Roles?
10. What is a hypervisor and which hypervisors are supported in OpenStack?
11. Explain the meaning of ‘flavor’ in OpenStack.
12. Create a new network of 64 IP addresses in the Network tab and enable DHCP for 32 of the IPs using either the GUI or the CLI.
13. Create a router that connects this new network with the existing “public network” using either the GUI or the CLI.
14. Start two instances with the Cirros image present that connects to the new network of 64 IPs using either the GUI or the CLI.

# Objective 2 – Auto-scaling application using Python

1. Scenario:

You are working in a cloud firm that has a single instance of an application running on OpenStack cloud platform. The firm is planning to add a functionality to the single running instance of the application that can autoscale/replicate itself to multiple instances whenever the compute capacity (eg. CPU cycles or memory) reaches a pre-defined threshold. Since you are familiar with the Python programming and REST API, you are being assigned a following task:

* 1. Write a simple Python application that can ssh into the available “cirros” instance that was created in the above objective and extract the CPU utilization information. [As an alternative, you may use ceilometer service for retrieving this telemetry data]
  2. If the CPU utilization or memory usage exceeds a threshold value, for example 20%, spin up additional instances of cirros. The creation of cirros instances should be triggered whenever the usage of CPU or memory exceeds a predefined threshold. Select CPU or memory usage to your interest to define your condition to trigger the creation of additional instances. In order to collect the utilization data, you’ll have to monitor its usage using appropriate commands.
  3. The Python application can use Nova REST API to create additional “Cirros” instances whenever the above condition occurs.
  4. The auto scaling of the instances should be handled considering following requirements:

**Max scaling size: 4** (this value denotes the maximum number of instances that should be spun)

**Increment size: 1** (this value denotes the number of instances that should be spun whenever CPU utilization exceeds threshold)

**Evaluation period: 40** (this value denotes the time period in seconds for monitoring CPU usage)

1. You can use the [Linux stress tool](https://www.tecmint.com/linux-cpu-load-stress-test-with-stress-ng-tool/) to raise the CPU utilization of an instance above the threshold.

# Objective 3: Multi-tenants

* In this objective, you are introduced to the function of basic tenant implementation and management with OpenStack.
* The goal is to create two virtual networks and three VMs as is shown in Figure 1.

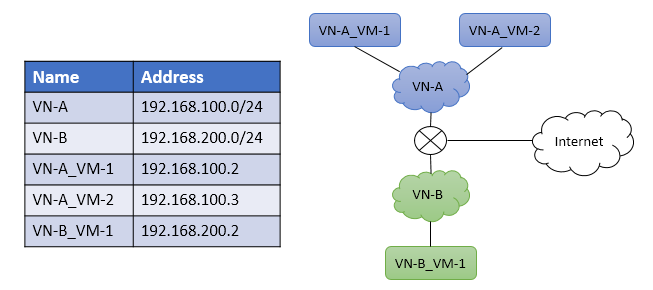


Figure 1. Final goal of Objective 3

# Section 1: Creating project, user, flavor and image

1. Within OpenStack UI Identity tag, create a project called lab2. Then create a user called lab2\_admin and attach it to the project lab2.
2. Within OpenStack UI Admin tag, create a VM Flavor called **ngn.tiny** with the following setting (or the setting that works for your VM image):

vCPU = 1

RAM = 128MB

Root Disk = 1GB

Ephemeral Disk = 1GB

Swap Disk = 1GB

1. Within OpenStack UI Admin tag, upload a VM image into OpenStack. You can use this URL: <http://tinycorelinux.net/7.x/x86/release/Core-current.iso> or <https://docs.openstack.org/image-guide/obtain-images.html>.

Remember to make it public.

1. Before proceeding, logout and login with your newly created user lab2\_admin.

# Section 2: Setup Virtual Networks

1. Login back into OpenStack UI, within the Project tag, create a new Network called VN-A with network address 192.168.100.0/24.
2. Repeat the above steps to create a second network VN-B with network address 192.168.200.0/24.

# Section 3: Launch VM instances

Launch the following VMs using the flavor and image created in Section 1.

1. Launch VN-A\_VM-1 and VN-A\_VM-2 into virtual network VN-A.
2. Launch VN-B\_VM-1 into virtual network VN-B.

# Section 4: Ping testing

1. Use the console within OpenStack UI to test if VMs in VN-A can ping each other, while the VM in VN-B cannot reach VMs in VN-A.
2. Assign floating IP’s to the VM’s both in VN-A and VN-B, and test connectivity to the Internet.

# Objective 4 – Network policies in OpenStack

# Summary:

In this objective, you will manage network policies within OpenStack.

# Section 1: VM and Virtual Network Setup

First you will need to create the configuration as is shown in Figure 1 above.

# Section 2: Achieve Inter-VN Communication

The default policy allows only intra-VN communication.

1. Figure out how to ping between VM’s in VN-A and VM’s in VN-B.
2. Figure out how to ping from the VM’s out to the Internet.

# Section 3: Network Policy Management

Now manage the network policy inside OpenStack, such that:

1. VN-A\_VM-1 can ping VN-B\_VM-1, but VN-A\_VM-2 cannot ping VN-B\_VM-1.
2. VN-B\_VM-1 can go out to the Internet, but VN-A\_VM-1 and VN-A\_VM-2 cannot.

# Deliverable (100 points):

## Individually complete all tasks in the lab

## Create a capstone group GitHub tutorial document about this lab

## Create a small guide explaining/demonstrating how to achieve each objective from the lab

## The individual objective guides should be divided between the team members evenly to be completed in GitHub

## Submit to Canvas:

## Individual Python Auto-scaling application

## Document (bulleted list) of what each member contributed to GitHub tutorial document

## Include a link to the GitHub page